SyncTwin: Fast Digital Twin Construction and Synchronization for Safe Robotic Grasping

Anonymous CVPR submission

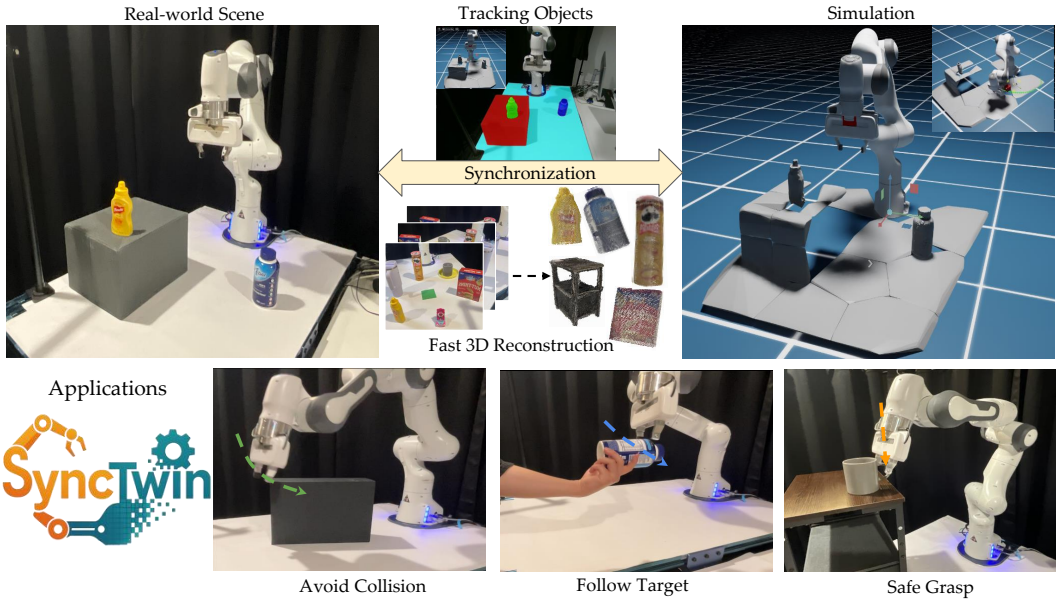

Figure 1. SyncTwin enables fast digital twin and synchronization based on efficient 3D scene reconstruction and object tracking, bridging motion planning for sim-to-real execution, which can be applied to dynamic obstacle avoidance, object tracking, and safe grasping under single-view occlusion in real-world environments.

Abstract

Accurate and safe grasping under dynamic and visually occluded conditions remains a core challenge in real-world robotic manipulation. We introduce SyncTwin, a novel digital twin framework that unifies fast 3D scene reconstruction and real-to-sim synchronization for robust and safety-aware grasping in such environments. In the offline stage, we employ VGGT to rapidly reconstruct object-level 3D assets from RGB images, forming a reusable geometry library for simulation. During execution, SyncTwin continuously synchronizes the digital twin by tracking real-world object states via point cloud segmentation updates and aligning them through colored-ICP registration. The updated twin enables motion planners to compute collision-free and dynamically feasible trajectories in simulation, which are safely executed on the real robot through a closed real-to-sim-to-real loop. Experiments in dynamic and occluded scenes show that SyncTwin improves grasp accuracy and motion safety, demonstrating the effectiveness of digital-twin synchronization for real-world robotic execution.

1. Introduction

Achieving accurate and safe robotic grasping in dynamic real-world environments remains a long-standing challenge, due to incomplete perception and dynamic scenes. Without an accurate understanding of their surroundings, robots risk colliding with the environment, which may damage hard-

ware or even endanger humans [58]. Thus, ensuring that robots can safely plan and execute motions amid dynamic scene changes is a prerequisite for reliable real-world deployment. Unlike simulation, where complete scene geometry and object states are fully accessible, real-world perception often only offers partial, occluded observations. As a result, motion-planning algorithms [16, 33, 36] that rely on full and accurate environment models in simulation face significant challenges when deployed in the real world.

Several recent efforts have attempted to bridge the gap between perception and control. Voxel-based mapping systems like NVBlox [23] enable online obstacle reconstruction, yet they only provide a voxel grid, which is a representation too coarse for reliable manipulation. End-to-end reactive policies like DRP [53] handle dynamics through continuous control but require robot-specific retraining and lack generalization across new hardware or environments. Thus, all of these methods share a critical limitation: they operate without a consistent and complete model of the scene, leading to unsafe or unreliable executions in the real world.

Bridging this gap between simulation and the real world, a robot needs a perception model that can efficiently perceive what objects exist in the environment and also track how the real-world scene evolves over time. In other words, instead of planning in a static or outdated simulation, the robot should plan within a dynamic digital twin that mirrors the physical world in real time, where perception and control are tightly coupled through continuous real-to-sim synchronization. However, real-world perception is inherently partial—single-view occlusions often reveal only fragments of object geometry, making grasping and motion planning unreliable. Inspired by SAM4D [51], which maintains a memory bank of object assets, we incorporate the idea of leveraging object-level memories to complete partial observations at execution time. We develop **SyncTwin**, a digital-twin framework equipped with an object memory bank that performs real-time object tracking from point clouds, injects accurate poses and geometries into simulation, and closes the loop by executing planned trajectories back on the real robot, as illustrated in Figure 1.

SyncTwin operates in two stages. In Stage I, VGGT [46] generates scene-level point clouds from RGB inputs. Object-level point clouds are extracted via projection, segmentation, and denoising, then converted into lightweight meshes and stored in a memory bank. In Stage II, the digital twin is continuously synchronizes with the real world using SAM2 for object tracking and GPU-accelerated colored-ICP [42]. Combined, these two methods align partial observations with stored assets to maintain a consistent and complete scene representation. The updated digital twin is streamed into Isaac Sim [24], allowing cuRobo [41] to perform motion planning and generate collision-aware trajectories. Moreover, this architecture can be transferred to different real robots without any retraining.

The main contributions of this paper are as follows:

- We present the first digital twin framework that tracks 3D objects in real time from point clouds and updates their poses and geometries in a synchronized simulation, enabling collision-aware planning and a closed real-to-sim-to-real loop for dynamic, partially observed scenes.
- We introduce a fast, low-cost RGB-only method that constructs 3D geometry assets using learning-based geometry estimation and projection-based segmentation.
- We develop a real-time 3D segmentation and tracking module that processes streaming RGB-D camera data.
- Our experiments demonstrate that the proposed system improves grasp accuracy and safety in single-view and occluded settings, while also achieving state-of-the-art efficiency in 3D geometry asset reconstruction.

2. Related Work

3D Scene Reconstruction and Segmentation. Traditional 3D scene reconstruction approaches rely on RGB-D input from depth sensors to estimate geometry [5, 7, 28, 38]. More recent approaches operate purely on RGB images, either through optimization-based pipelines such as Structure-from-Motion (SfM) [14, 30, 31, 35] and Multi-View Stereo (MVS) [10, 11, 37], or through learning-based frameworks such as MVSNet [54], NeRF [22], and 3D Gaussian Splatting [20]. While these methods can recover visually plausible geometry, they often require substantial computation time to produce complete reconstructions. For robotics applications, object-centric 3D segmentation is important as well. Prior work has explored segmentation directly on point clouds using learned clustering and aggregation strategies [15, 47, 57], as well as multi-view projection of 2D masks for 3D instance segmentation [21, 34, 56]. However, projection-based methods typically require accurate camera extrinsics [2, 52], and little research has examined how such strategies perform when applied to point clouds produced by learning-based reconstruction methods. Our work addresses this gap by unifying learned 3D reconstruction with mask-guided multi-view segmentation, enabling efficient and robust object asset generation for downstream robotic manipulation.

Digital Twin for Robotic Manipulation. Digital twin systems have become increasingly popular for sim-to-real transfer, particularly in training reinforcement learning and imitation learning policies [4, 45, 50, 55]. And several studies leverage generated digital twins as a form of data augmentation to enhance the generalization of downstream models [19, 25]. However, existing frameworks typically reconstruct static scenes once before training [29] and do not maintain continuous synchronization with the evolving physical world. Some recent efforts attempt real-time tracking by detecting object locations with 2D detectors [40], but

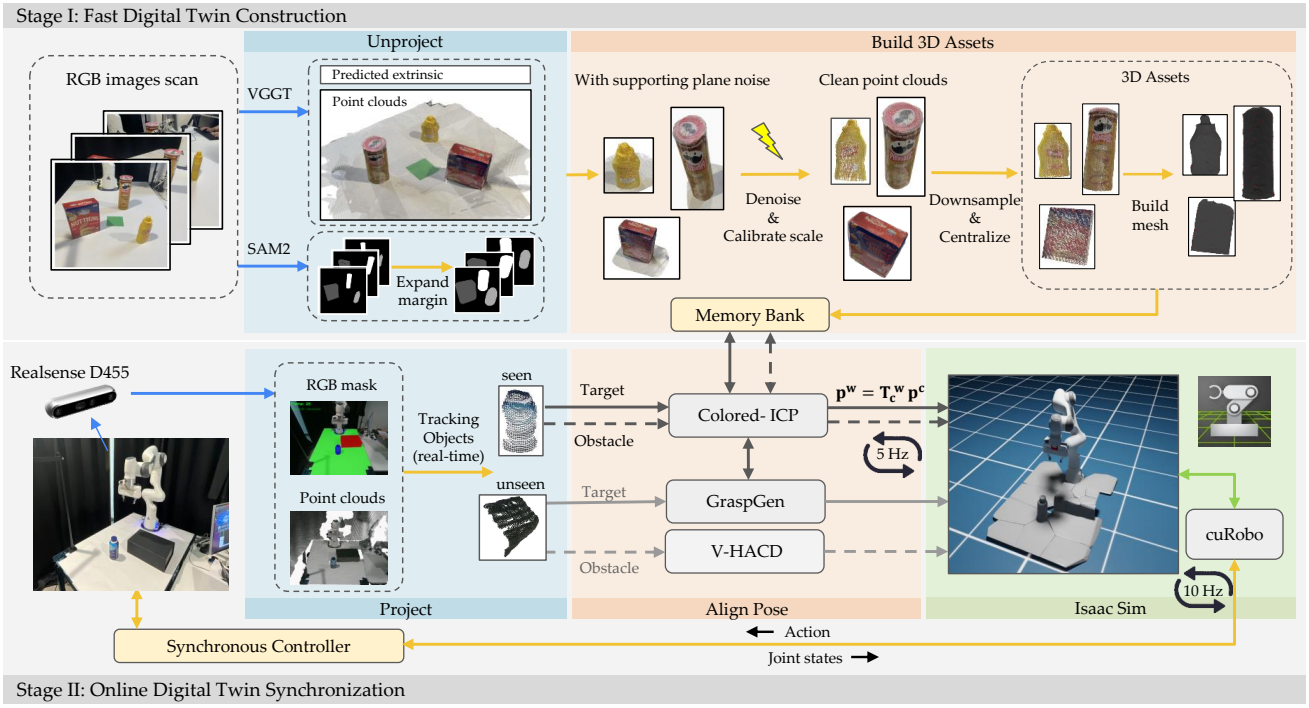

Figure 2. **Framework of the SyncTwin**. Stage I reconstructs simulation-ready 3D assets from RGB images using VGGT and SAM2. Multi-view masks are unprojected into point clouds, then denoised, scaled, and meshed into clean object assets stored in a memory bank. Stage II performs real-time object segmentation, pose tracking, and asset-based completion, enabling grasp generation and reactive motion planning in a closed real2sim2real loop. By continuously updating the digital twin and leveraging simulation for decision making, the system ensures safe and adaptive execution under dynamic and partially occluded environments.

they often lack precise pose estimation or geometry updates. In contrast, our system enables continuous synchronization of the digital twin with real-world online perception, which enables accurate tracking and reliable manipulation in dynamic environments under occlusion.

Safe Motion Planning for Manipulation. Safe and robust motion planning remains a critical challenge for robotic manipulation in unstructured environments [18, 39]. Classical and learning-based planners [16, 33, 36] typically operate in simulation, where the environment is fully accessible. When deployed in the real world, however, they must handle partial observations. To narrow this gap, several methods construct static point cloud maps and import them into simulation for offline planning [3], though such maps struggle to support real-time adaptation. Other approaches aim to ensure safety by predicting collision-free actions directly from images in latent space [1, 27], using force-sensing-based control [17, 48], or distilling planning policies from point cloud observations [6, 9, 53]. NVBlox improves online safety by voxelizing scene geometry and segmenting the robot in real time [23]. In contrast, our method maintains a dynamically synchronized digital twin that continuously provides updated scene geometry to the planner, enabling safe execution in dynamic, cluttered environments.

3. Method

SyncTwin consists of two stages: (1) fast digital twin construction, and (2) digital twin synchronization. An overall framework is provided in Figure 2.

3.1. Problem Formulation

We aim to enable safe robotic grasping in dynamic, partially observable real-world environments by maintaining a *continuously synchronized digital twin*. This problem can be decomposed into the following components:

Stage I: Fast Digital Twin Construction. The system receives a small set of RGB images $\{\mathbf{I}_i\}_{i=1}^N$ along with camera intrinsics K and estimated extrinsics $\{\mathbf{T}_i^{\text{world}}\}_{i=1}^N$. The goal is to produce object-level, simulation-ready 3D assets $\mathbb{B} = \{\mathcal{X}_j, \mathcal{M}_j^{3D}\}$ from these images. The main challenge is that learning-based extrinsics contain unstable errors, causing mask–projection misalignment and table–object mixing in the reconstructed point cloud, which must be addressed to obtain clean object geometry.

Stage II: Online Digital Twin Synchronization. During execution, the system receives streaming RGB-D frames and corresponding partial point clouds $\mathcal{X}_p$. The objective is to maintain accurate object poses $\mathbf{T}_j^{\text{world}}$ in the simula-

tor by aligning $\mathcal{X}_p$ with their complete assets $\mathcal{X}_m \in \mathbb{B}$. This synchronized scene is streamed into Isaac Sim [24], where cuRobo's MPC planner [41] produces short-horizon, collision-free trajectories $\mathbf{A}_{t:t+H} = \{\mathbf{a}_0, \ldots, \mathbf{a}_H\}$. The key challenge is robustly tracking objects under occlusion and partial observation, while ensuring that the object poses and geometries can be accurately updated into the simulator to enable safe real-to-sim-to-real planning.

3.2. Fast Digital Twin Construction

The first stage aims to rapidly reconstruct the 3D environment and extract object-level representations suitable for real-time simulation, where the focus is to achieve accurate geometric perception for motion planning, rather than photorealistic reconstruction. We employ VGGT [46] to reconstruct dense scene point clouds directly from a small number of RGB images, which enables efficient extraction of object-level geometry in a fast and low-cost manner without depth sensors or multi-view optimization.

Nevertheless, there are two major practical challenges. First, the camera extrinsics estimated by VGGT are often inaccurate, which leads to projection misalignment during mask-based segmentation. Second, the generated point cloud is not in a true metric world scale, causing the imported assets to appear incorrectly sized relative to the robot in the simulator. Therefore, Stage I focuses on producing simulation-ready 3D digital assets from VGGT outputs by addressing these limitations. We design a four-step pipeline that corrects extrinsic inaccuracies, enforces scale consistency, and generates clean object meshes suitable for downstream planning, which is introduced as follows.

Mask Projection Expansion. To mitigate inaccuracies in VGGT-estimated camera extrinsics, each 2D segmentation mask $\mathcal{S}_i$ is spatially expanded before projection, which ensures full coverage of object boundaries and prevents missing edge regions during 3D reconstruction. While mask expansion compensates for projection drift, it also introduces floating outliers (e.g., background points above the object) and merged support-plane regions (e.g., table surfaces). To address this, we apply our point clouds denoising mechanism to isolate the true object shape.

Point Clouds Denoising. Detecting openings or cavities in 3D point clouds is a fundamental step in shape understanding for denoising table noise. We propose a purely geometric method that progressively expands a virtual light sphere from the object's center and tracks uncovered regions on the spherical sampling space. The algorithm automatically detects openings and extracts rim points around their boundaries without requiring mesh topology or prior segmentation. Figure 3 shows the overall process.

Given a point cloud $\mathcal{P}$, we first estimate a geometric center $\mathbf{c} = \mathrm{mean}(\mathcal{P})$. We then discretize the unit sphere into F directions $\{\mathbf{d}_i\}_{i=1}^{F}$ using a Fibonacci spiral distribution [13], forming a uniform sampling domain $\mathcal{D}$. Each point $\mathbf{p}_i$ defines a normalized direction $\hat{\mathbf{v}}_i = (\mathbf{p}_i - \mathbf{c})/\|\mathbf{p}_i - \mathbf{c}\|$ and is assigned to its nearest angular bucket direction $\mathbf{d}_j$ if $\hat{\mathbf{v}}_i \cdot \mathbf{d}_j \geq \cos(\theta_{\text{tolerance}})$, We then iteratively expand a virtual sphere centered at $\mathbf{c}$ with radius r_t:

$$r_{t+1} = r_t + \Delta r, \quad r_{\min}(j) = \min_{i \in \text{bucket } j} \|\mathbf{p}_i - \mathbf{c}\|. \tag{1}$$

A direction $\mathbf{d}_j$ is marked as *hit* once any assigned point enters the sphere, and unhit directions form a binary mask $\mathcal{U}_t$. Stable uncovered regions (openings or cavities) are detected when the largest unhit component remains consistent over iterations. We denote by $\mathcal{N}(j) = \{k \mid (j,k) \in \mathcal{P}\}$ the neighborhood of bucket j. Boundary buckets are defined as unhit directions adjacent to hit ones:

$$\mathcal{B} = \{j \in \mathcal{U} \mid \exists\, k \in \mathcal{N}(j), \mathrm{hit}(k) = 1\}. \tag{2}$$

For each boundary $\mathbf{d}_j$, the farthest point within tolerance is chosen as a rim sample $\mathbf{p}_j^*$. Connected components on the spherical adjacency graph are extracted to identify large uncovered regions. Let $\{\mathcal{C}_m\}$ denote all connected components of the unhit set $\mathcal{U}_t, \mathcal{C}_{\max}$ is the largest connected region. The principal opening direction is then computed by averaging the largest unhit component: $\mathbf{n}_{\text{open}} = \frac{\sum_{j \in \mathcal{C}_{\max}} \mathbf{d}_j}{\| \sum_{j \in \mathcal{C}_{\max}} \mathbf{d}_j \|}$. Finally, the rim points $\{\mathbf{p}_j^*\}$ are fitted with a plane using SVD [12], yielding the opening orientation and visualizable boundary. Compared with RANSAC plane fitting [8], which can only segment dominant planes, our method can detect cavity openings, enabling the identification of ring-shaped planes surrounding the opening

Real-World Scale Alignment. Since VGGT produces point clouds that are not in a world-scale metric, we estimate a global scale factor to align the reconstructions with world coordinates. Even with known intrinsic parameters, according to the pinhole camera model, monocular geometry cannot determine absolute scale [14]. Therefore, we

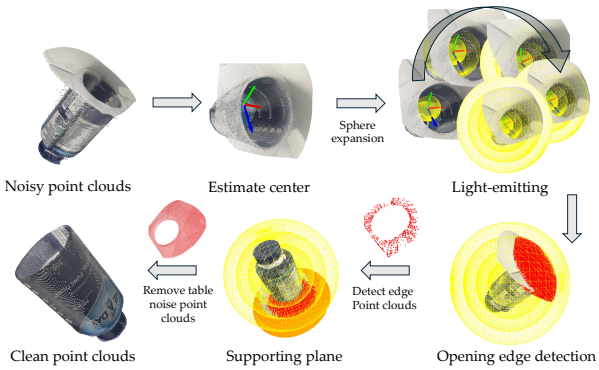

Figure 3. **Supporting-plane noise removal mechanism.** A virtual light sphere expands from the object center to identify openings and boundary points, enabling filtering of table-plane noise.

calibrate the scale using a reference object or markers of known physical dimensions within the scene. Implementation details are provided in supplementary.

Mesh Simplification. To maintain real-time performance in the digital twin, we apply an adaptive mesh decimation that reduces vertex count while preserving geometric fidelity and collision boundaries. To avoid over-smoothing across sharp edges, we use an angle-based gating weight $w_{ij} = 1$ if $\theta_{ij} \leq \theta_{\text{th}}$ and 0 otherwise, where θ_{th} is a feature threshold (e.g., 30°) and $\theta_{ij} = \arccos(\mathbf{n}_i^\top \mathbf{n}_j)$ for all $j \in \mathcal{N}(i)$. Each vertex is then updated via a selective Laplacian step [44]. Compared to uniform mesh decimation, this feature-aware smoothing preserves sharp edges around handles and object rims, which are critical for accurate grasp planning. All processed point clouds, meshes, and their id are stored in a memory bank, which serves for object recognition and scene synchronization in Stage II.

3.3. Online Digital Twin Synchronization

The second stage of our system focuses on real-time object tracking and safe grasp execution through continuous perception, planning synchronization between the real and digital environments. This stage consists of four tightly integrated modules: real-time point cloud segmentation, GPU-accelerated colored-ICP registration, grasp pose generation from complete object models, and dynamic motion planning with cuRobo MPC.

Real-time Point Clouds Segmentation. To achieve real-time segmentation on incoming RGB-D streams, we build a module shown in Fig 4, that performs continuous inference on camera frames and outputs segmentation masks $\mathcal{S}_p$, and then projects the mask onto the full point cloud to obtain the corresponding partial object point clouds $\mathcal{X}_p$. Compared with traditional offline SAM2 inference, we design a sliding window mechanism that enables SAM2 to process camera streams in real time, maintaining temporal consistency in object masks for continuous 3D segmentation and tracking under occlusions. At image size 640×480, our method runs at 15 Hz on RTX 4090.

Colored-ICP Registration. For aligning partial point clouds $\mathcal{X}_p$ obtained from the camera with the corresponding full object model $\mathcal{X}_m$ stored in the memory bank, we employ a colored-ICP algorithm [32] implemented on the GPU via the cupoch library [42]. Unlike traditional geometric ICP, which minimizes only spatial distance, colored-ICP jointly minimizes geometric and color residuals: $E(R,t) = \sum_i [\lambda_g \|R\mathbf{x}_i + t - \mathbf{y}_i\|^2 + \lambda_c \|I(\mathbf{x}_i) - I(\mathbf{y}_i)\|^2]$ where (R,t) denotes the transformation between $\mathcal{X}_p$ and $\mathcal{X}_m$, and $I(\cdot)$ represents point color intensity. The weighting factors λ_g and λ_c balance geometric and color terms.

Grasp Pose Generation and Obstacle Representation. For unseen target objects, we directly apply GraspGen to the partial point cloud. In contrast, once a seen target object is registered, we replace its partial observation with the complete point cloud $\mathcal{X}_m$ from the memory bank and feed it into GraspGen to predict grasp poses $\{\mathbf{T}_{\mathbf{gripper}} = f_{\text{GraspGen}}(\mathcal{X}_m)\}$,This replacement mitigates the uncertainty from occlusion and single-view perception, yielding more stable and accurate grasp pose estimation. For unseen obstacles, we dynamically generate multi-convex hulls (V-HACD [49]) from the segmented point cloud for collision modeling. For known obstacles, the aligned object meshes and poses are directly imported into the digital twin for real-time collision checking.

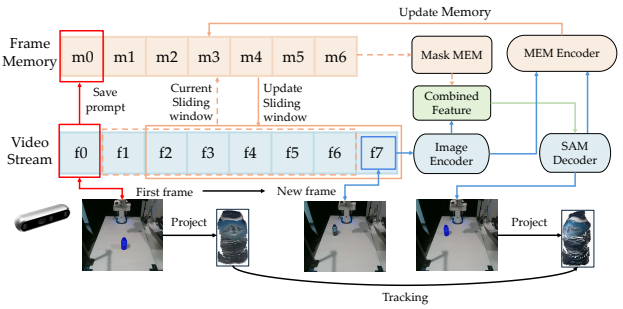

Figure 4. **Overview of the camera predictor module**. The red solid line indicates that the first frame is persistently stored in the frame memory. The yellow dashed line represents the operation of saving frames from the previous time step into the memory. The blue and green solid lines denote the data flow and processing steps for the current frame. Together, the sliding-window mechanism enables real-time video segmentation and object-level point cloud tracking with temporally consistent memory updates.

Motion Planning with Sim-to-real Synchronization. Finally, motion planning and control are performed using the cuRobo's model predictive control (MPC) framework. At each control step, the robot's joint states are synchronized with the digital twin, where cuRobo computes an optimized short-horizon trajectory under real-time collision constraints. Only the first control action $\{\mathbf{a}_0\}$ from the predicted trajectory $\mathbf{A}_{t:t+H} = \{\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_H\}$ is executed on the real robot, followed by continuous replanning with the environment updates, achieving closed-loop synchronization between simulation and reality.

4. Experiments

We evaluate SyncTwin across three dimensions: (1) the efficiency of the fast 3D reconstruction pipeline; (2) obstacle avoidance performance under dynamic and single-view occluded conditions; and (3) grasp success rate under single-view occlusion. All experiments are designed to validate both the offline and online components of our framework.

4.1. Experiment Setup

All experiments are conducted with a Franka Emika Panda robotic arm equipped with an Intel RealSense D455 RGB-

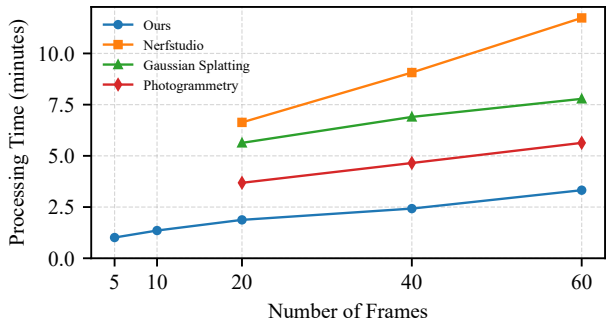

Figure 5. **The comparison of processing time given different numbers of input images.** The processing time covers both reconstruction and segmentation. The 5 and 10 images do not apply to the baselines because of fail to estimate the camera extrinsic.

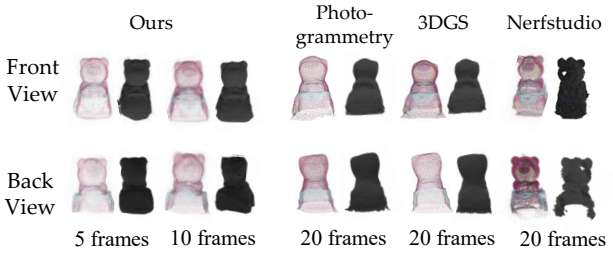

Figure 6. **Reconstruction comparison given different numbers of input images**. In each column, the left figure shows the point clouds, and the right one shows the untextured mesh.

D camera mounted above and in front of the workspace. And iPhone 12 for RGB images. We apply voxel-based downsampling to the input point cloud with a voxel size of 3 mm. This is motivated by the fact that the computational complexity of point cloud segmentation algorithms scales linearly with the number of points. The digital twin is implemented in Isaac Sim 4.0 and integrated with the cuRobo MPC motion planning framework, running on a single NVIDIA RTX 4090 GPU. Both perception and planning run on the same GPU, enabling a closed-loop update rate of up to 5 Hz. Motion planning runs at 10 Hz, and the robot's velocity scaling is set to 0.2. The test objects include bottles, cans, cups, and boxes of various shapes.

4.2. Baselines and Metrics

Baselines. (1) For *3D reconstruction*: we compare against Photogrammetry [35], NeRF (Nerfstudio [43]), and Gaussian Splatting (3DGS [20]). (2) For *obstacle avoidance*: we adopt NVBlox [23] as the baseline voxel-mapping method. For the ablation studies, we analyze performance across the following settings: (1) *Mask Expansion and Denoising*: using variants without mask expansion and without denoising as baselines, evaluating their reconstruction quality against our full segmentation mechanism. (2) *Object Completion*: we use GraspGen [26] as the baseline grasp generator, comparing grasp poses predicted from single-view partial point clouds (baseline) versus from complete, asset-retrieved geometry produced by SyncTwin (ours).

Evaluation Metrics. Depending on the experiment, we report the following metrics: reconstruction time (min), dependency on the number of input images $N_{\min}$, obstacle avoidance success rate (%), and grasp success rate (%). For avoidance tests, we define 3-level outcome levels: FA (full avoidance, no contact with the object, 1.0), EA (edge avoidance, slight contact without displacing the object, 0.8), and CO (collision, noticeable contact that significantly moves the object, 0.0). Weighted success rate:

$$\text{SR} = \frac{N_{\text{FA}} + 0.8\,N_{\text{EA}} + 0.0\,N_{\text{CO}}}{N} \times 100\%.$$

4.3. Fast 3D Reconstruction

We evaluate the efficiency and input-image dependency of SyncTwin's 3D reconstruction module, comparing against Photogrammetry, 3DGS, and Nerfstudio. Each method reconstructs the same object using 5, 10 ,20, 40, 60 RGB images, and all approaches obtain object assets through multi-view projection based segmentation. The experimental results are summarized as follows.

Reconstruction Time. As shown in Figure 5, SyncTwin achieves the shortest reconstruction time across all settings. It produces a simulation-ready mesh in only about 1–2 minutes using 5–10 input images, while Photogrammetry, 3DGS, and Nerfstudio require at least 4–7 minutes even with more frames. This enables significantly faster digital-twin construction in a new real-world scenarios.

Dependency on the Number of Input Images. SyncTwin also exhibits the lowest dependency on input-image count. It generates usable meshes from as few as 5–10 images, whereas competing approaches typically require 20+ images to avoid failure caused by unstable optimization of camera extrinsics. This low image dependency substantially accelerates asset generation and improves robustness under limited views, especially because the reported processing time excludes image-capture time, which becomes additional and unpredictable when more images are required.

Furthermore, the qualitative comparison in Figure 6 shows that our method preserves fine-grained geometric details even with very few input images, as our mesh simplification algorithm maintains high geometric fidelity while reducing mesh vertices, enabling faster simulation performance. For instance, the *bear's ear shape* remains well preserved with only 5–10 frames, demonstrating both low image dependency and strong geometric consistency.

These properties make SyncTwin a state-of-the-art solution for fast 3D geometry asset generation: the system

Table 1. **Comparison of obstacle-avoidance performance between NVBlox and SyncTwin in dynamic environments**. Unseen objects (left) are not present in the asset memory, whereas starred objects in the Seen category (right) are stored in the memory bank. SyncTwin achieves significantly higher success rates in both settings, with particularly strong gains when complete object assets are available.

Method	Motion	Unseen													Seen (Memory Bank)												
		Box1			Box2			Box3			Box4			SR_unseen (%)	Box1*			Box2*			Box3*			Box4*			SR_seen (%)
		FA	EA	CO	FA	EA	CO	FA	EA	CO	FA	EA	CO		FA	EA	CO	FA	EA	CO	FA	EA	CO	FA	EA	CO	
NVBlox	SelfRot	6	4	10	9	5	6	3	4	13	3	11	6	50.3%	–	–	–	–	–	–	–	–	–	–	–	–	–
	EnterTraj	3	9	8	4	9	7	1	6	13	0	3	17	37.0%	–	–	–	–	–	–	–	–	–	–	–	–	–
SyncTwin	SelfRot	12	7	1	14	5	1	11	7	2	9	9	2	**85.5%**	18	1	1	16	4	0	15	5	0	13	6	1	**93.5%**
	EnterTraj	8	8	4	11	6	3	8	7	5	7	8	5	**71.5%**	12	7	3	12	5	3	10	6	4	9	7	4	**78.8%**

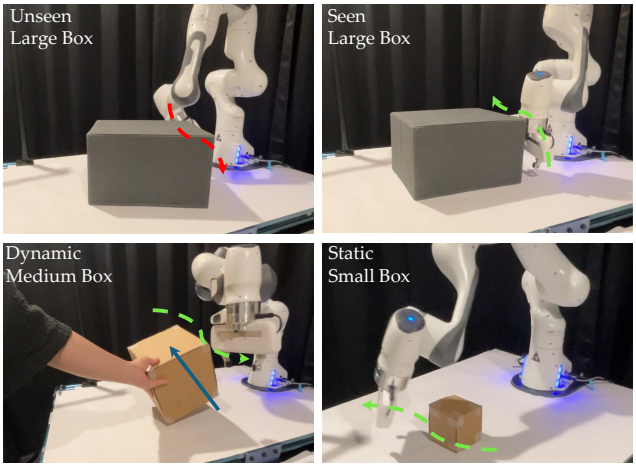

Figure 7. **Examples of SyncTwin's dynamic obstacle avoidance**. Green dashed lines indicate collision-free robot trajectories, while red lines mark trajectories that result in collisions. Blue arrows denote the motion of dynamic obstacles. For unseen objects (top-left), the robot collides with unobserved regions, whereas the same object in the seen case (top-right) is successfully avoided. SyncTwin also handles dynamic obstacles (bottom-left) and small objects (bottom-right) effectively.

can construct high-quality, simulation-ready object meshes in about 1 minute, enabling rapid reconstruction of object assets for downstream digital-twin synchronization.

4.4. Obstacle Avoidance under Occlusion

We adopt the built-in obstacle avoidance benchmark in cuRobo to evaluate the success rate of obstacle avoidance, where the robot repeatedly moves between two target points while avoiding obstacles along its path.

Both NVBlox and SyncTwin are tested on unseen (not stored in memory bank) obstacles to evaluate avoidance performance. Additionally, SyncTwin is evaluated on seen (stored in memory bank) objects recorded in its asset memory to study how prior geometry improves safety and reactivity under single-view occlusion. The experiment tests under two motion patterns: **SelfRot** indicates in-place rotation; **EnterTraj** blocks motion into the predicted trajectory. Each condition is repeated for 20 trials, total $N = 20 \times 4 = 80$. We test four boxes with sizes of Box1: $10\times10\times10$ cm, Box2: $20\times20\times20$ cm, Box3: $10\times20\times30$ cm, and Box4: $20\times22\times35$ cm as obstacles. Through experimental observation, the results are as follows:

Unseen Object Performance. As shown in Table 1, SyncTwin consistently outperforms NVBlox when encountering unseen obstacles under single-view dynamic occlusion. NVBlox exhibits failure modes such as: (1) unstable voxelization for small objects (e.g., Box1) due to sparse depth returns; (2) misclassification of limited height objects (e.g., Box3) as part of the tabletop caused by inconsistent depth estimation; and (3) trajectory intersections when large obstacles (e.g., Box4) fall outside the sensor's visible region. Across all these cases, SyncTwin maintains markedly higher obstacle-avoidance success rates, demonstrating stronger robustness to object scale, occlusion, and limited viewpoint coverage.

Seen Object Performance. In Table 1, when objects are stored in the asset memory, SyncTwin's performance improves even further. Complete geometric priors obtained in Stage I resolve the challenges posed by small or thin objects, eliminate many failure modes, and convert numerous edge-avoidance cases into full avoidance. These results indicate that SyncTwin not only generalizes better on unseen objects but also achieves substantially higher reliability when the objects have been previously reconstructed. These results highlight the importance of SyncTwin's Stage I memory construction, which enables complete geometric reasoning even under partial observability. Furthermore, Figure 7 illustrates representative avoidance examples achieved by SyncTwin.

4.5. Ablation Studies

Mask Expansion and Denoising. Due to the inaccuracy of VGGT-estimated camera extrinsics, multi-view mask projections often become misaligned and fail to fully cover the object. Our ablation study evaluates reconstruction results *w/o mask expansion* and *w/o denoising*. As shown in

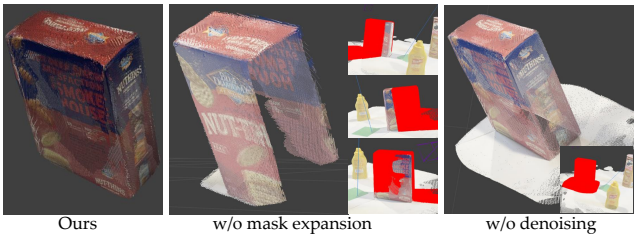

Figure 8. **Comparison of segmentation-based 3D reconstruction results**. Left: Our method generates a complete and clean object mesh without support-plane noise. Middle: w/o mask expansion, multi-view projections (red) fail to fully overlap, and their intersection leads to missing object geometry. Right: w/o denoising, the intersection of projected regions (red) incorrectly preserves support-plane points, introducing significant noise into the reconstructed point cloud.

Table 2. **Comparison of grasp success rates before and after geometry completion**. Asset-based completion significantly improves performance across all objects by producing more accurate and safer grasp pose candidates, with the largest gains observed for partially occluded items such as the cup (with handle).

Condition	Bottle	Cup (Handle)	Cookie Box	Chips Can
Before Completion (%)	78.3	65.0	81.7	80.0
After Completion (%)	**90.0**	**86.7**	**93.3**	**95.0**
Improvement	+11.7	+21.7	+11.6	+15.0

Figure 8, removing margin expansion causes inconsistent projections across viewpoints, leaving parts of the object missing. Likewise, disabling supporting-plane points denoising preserves table artifacts in the point cloud, resulting in noisy meshes. These comparisons demonstrate that segmentation-aware expansion and support-plane denoising are essential for obtaining clean and stable 3D assets.

Object Completion. We evaluate the effect of object completeness on grasp generation. When grasp poses are generated from single-view partial point clouds, the limited geometry often leads to incomplete or incorrect grasp configurations, for example. As shown in Figure 9, the cup handle is misinterpreted, resulting in unsafe or colliding grasps. In contrast, after SyncTwin aligns the observed object with the corresponding complete mesh from the asset library, the grasp generator produces denser, more accurate, and physically feasible grasp candidates. This demonstrates that asset-based completion is essential for reliable and collision-free grasp execution in the real world. As shown in Table 2, across 60 evaluation trials, grasp success rates increase substantially after geometry completion. The improvement is most pronounced for partially occluded or asymmetric objects, such as the cup and chips can, where asset-based completion provides the missing structure needed for reliable planning.

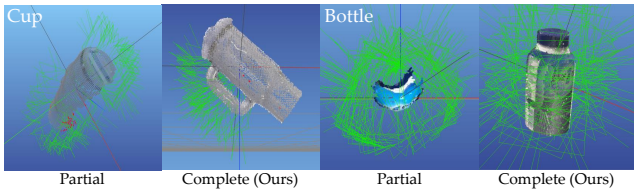

Figure 9. **Comparison of grasp candidates generated from Ours (w/o Completion) and Ours.** The example objects include a handled cup (left pairs) and a bottle (right pairs). Green lines visualize predicted gripper poses.

4.6. Discussions

In our experiments, Stage I reconstruction occasionally fails when the object lacks a clear supporting surface or when the support plane is weakly connected to the object (e.g., a tall cup). In these cases, inaccurate camera extrinsics cause the 2D segmentation masks to be incorrectly projected into 3D, resulting in broken or incomplete object point clouds and missing geometry. Such extrinsic errors produced by learning-based reconstruction remain an open challenge.

5. Conclusion

We introduced SyncTwin, a digital-twin framework that unifies fast RGB-only 3D reconstruction with real-time scene synchronization for safe and robust grasping in dynamic, partially occluded environments. By leveraging VGGT-based reconstruction, segmentation-aware denoising, and memory-driven geometry completion, SyncTwin provides accurate object geometry and reliable grasp generation from limited visual input. By bridging the sim-to-real gap through consistent digital-twin updates, the system enables simulation-based planners to execute safe, collision-aware trajectories on real robots without retraining.

Our experiments demonstrate substantial improvements over existing baselines such as NVBlox, including higher obstacle-avoidance success rates, more stable behavior in a dynamic environment, and improved grasp performance under single-view occlusion. These results highlight the advantage of combining fast asset generation with persistent memory and real-time synchronization, enabling safe and accurate execution in real-world environments.

Looking forward, two extensions appear particularly promising. First, enabling online asset expansion by integrating Stage I construction into Stage II synchronization would allow the system to incrementally acquire new objects as they appear. Second, distributed or multi-GPU system designs that decouple perception, synchronization, and planning could further improve responsiveness under fast dynamic scenes. Consequently, these enable more adaptive, real-world–oriented digital-twin synchronization.

References

[1] Arpit Bahety, Arnav Balaji, Ben Abbatematteo, and Roberto Martín-Martín. Safemimic: Towards safe and autonomous human-to-robot imitation for mobile manipulation. *arXiv preprint arXiv:2506.15847*, 2025. 3

[2] Mohamed El Amine Boudjoghra, Angela Dai, Jean Lahoud, Hisham Cholakkal, Rao Muhammad Anwer, Salman Khan, and Fahad Shahbaz Khan. Open-yolo 3d: Towards fast and accurate open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2406.02548*, 2024. 2

[3] Lukas Brunke, Yanni Zhang, Ralf Römer, Jack Naimer, Nikola Staykov, Siqi Zhou, and Angela P Schoellig. Semantically safe robot manipulation: From semantic scene understanding to motion safeguards. *IEEE Robotics and Automation Letters*, 2025. 3

[4] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019. 2

[5] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4): 1, 2017. 2

[6] Murtaza Dalal, Jiahui Yang, Russell Mendonca, Youssef Khaky, Ruslan Salakhutdinov, and Deepak Pathak. Neural mp: A generalist neural motion planner. *arXiv preprint arXiv:2409.05864*, 2024. 3

[7] Mingsong Dou, Li Guan, Jan-Michael Frahm, and Henry Fuchs. Exploring high-level plane primitives for indoor 3d reconstruction with a hand-held rgb-d camera. In *Asian Conference on Computer Vision*, pages 94–108. Springer, 2012. 2

[8] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 4

[9] Adam Fishman, Adithyavairavan Murali, Clemens Eppner, Bryan Peele, Byron Boots, and Dieter Fox. Motion policy networks. In *conference on Robot Learning*, pages 967–977. PMLR, 2023. 3

[10] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 2

[11] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE international conference on computer vision*, pages 873–881, 2015. 2

[12] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear algebra*, pages 134–151. Springer, 1971. 4

[13] Álvaro González. Measurement of areas on a sphere using fibonacci and latitude–longitude lattices. *Mathematical geosciences*, 42(1):49–64, 2010. 4

[14] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2, 4

[15] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and Pattern recognition*, pages 4867–4876, 2020. 2

[16] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE international conference on robotics and automation*, pages 4569–4574. IEEE, 2011. 2, 3

[17] Jeon Ho Kang, Sagar Joshi, Ruopeng Huang, and Satyandra K Gupta. Robotic compliant object prying using diffusion policy guided by vision and force observations. *IEEE Robotics and Automation Letters*, 2025. 3

[18] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011. 3

[19] Pushkal Katara, Zhou Xian, and Katerina Fragkiadaki. Gen2sim: Scaling up robot learning in simulation with generative models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6672–6679. IEEE, 2024. 2

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2, 6

[21] John McCormac, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level slam. In *2018 international conference on 3D vision (3DV)*, pages 32–41. IEEE, 2018. 2

[22] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[23] Alexander Millane, Helen Oleynikova, Emilie Wirbel, Remo Steiner, Vikram Ramasamy, David Tingdahl, and Roland Siegwart. nvblox: Gpu-accelerated incremental signed distance field mapping. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2698–2705. IEEE, 2024. 2, 3, 6

[24] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, et al. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. 2, 4

[25] Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, et al. Robotwin: Dual-arm robot benchmark with generative digital twins. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 27649–27660, 2025. 2

[26] Adithyavairavan Murali, Balakumar Sundaralingam, Yu-Wei Chao, Wentao Yuan, Jun Yamada, Mark Carlson,

Fabio Ramos, Stan Birchfield, Dieter Fox, and Clemens Eppner. Graspgen: A diffusion-based framework for 6-dof grasping with on-generator training. *arXiv preprint arXiv:2507.13097*, 2025. 6

[27] Kensuke Nakamura, Lasse Peters, and Andrea Bajcsy. Generalizing safety beyond collision-avoidance via latent-space reachability analysis. *arXiv preprint arXiv:2502.00935*, 2025. 3

[28] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. 2

[29] Chuanruo Ning, Kuan Fang, and Wei-Chiu Ma. Prompting with the future: Open-world model predictive control with interactive digital twins. *arXiv preprint arXiv:2506.13761*, 2025. 2

[30] John Oliensis. A critique of structure-from-motion algorithms. *Computer Vision and Image Understanding*, 80(2): 172–214, 2000. 2

[31] Onur Özyeşil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion*. *Acta Numerica*, 26:305–364, 2017. 2

[32] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 143–152, 2017. 5

[33] Nathan Ratliff, Matt Zucker, J Andrew Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *2009 IEEE international conference on robotics and automation*, pages 489–494. IEEE, 2009. 2, 3

[34] Martin Runz, Maud Buffier, and Lourdes Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *2018 IEEE international symposium on mixed and augmented reality (ISMAR)*, pages 10–20. IEEE, 2018. 2

[35] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 2, 6

[36] John Schulman, Jonathan Ho, Alex X Lee, Ibrahim Awwal, Henry Bradlow, and Pieter Abbeel. Finding locally optimal, collision-free trajectories with sequential convex optimization. In *Robotics: science and systems*, pages 1–10. Berlin, Germany, 2013. 2, 3

[37] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, pages 519–528. IEEE, 2006. 2

[38] Frank Steinbrücker, Jürgen Sturm, and Daniel Cremers. Real-time visual odometry from dense rgb-d images. In *2011 IEEE international conference on computer vision workshops (ICCV Workshops)*, pages 719–722. IEEE, 2011. 2

[39] Ioan A Sucan, Mark Moll, and Lydia E Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012. 3

[40] Yuzhu Sun, Mien Van, Stephen McIlvanna, Nguyen Minh Nhat, Kabirat Olayemi, Jack Close, and Seán McLoone. Digital twin-driven reinforcement learning for obstacle avoidance in robot manipulators: A self-improving online training framework. *arXiv preprint arXiv:2403.13090*, 2024. 2

[41] Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. Curobo: Parallelized collision-free robot motion generation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8112–8119. IEEE, 2023. 2, 4

[42] Kenta Tanaka. cupoch – robotics with gpu computing, 2020. https://github.com/neka-nat/cupoch. 2, 5

[43] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 conference proceedings*, pages 1–12, 2023. 6

[44] Gabriel Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 351–358, 1995. 5

[45] Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *arXiv preprint arXiv:2403.03949*, 2024. 2

[46] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025. 2, 4

[47] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2569–2578, 2018. 2

[48] Lai Wei, Jiahua Ma, Yibo Hu, and Ruimao Zhang. Ensuring force safety in vision-guided robotic manipulation via implicit tactile calibration. *arXiv preprint arXiv:2412.10349*, 2024. 3

[49] Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3d meshes with collision-aware concavity and tree search. *ACM Transactions on Graphics (TOG)*, 41(4):1–18, 2022. 5

[50] Yuxuan Wu, Lei Pan, Wenhua Wu, Guangming Wang, Yanzi Miao, Fan Xu, and Hesheng Wang. Rl-gsbridge: 3d gaussian splatting based real2sim2real method for robotic manipulation learning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 192–198. IEEE, 2025. 2

[51] Jianyun Xu, Song Wang, Ziqian Ni, Chunyong Hu, Sheng Yang, Jianke Zhu, and Qiang Li. Sam4d: Segment anything in camera and lidar streams. *arXiv preprint arXiv:2506.21547*, 2025. 2

[52] Mutian Xu, Xingyilang Yin, Lingteng Qiu, Yang Liu, Xin Tong, and Xiaoguang Han. Sampro3d: Locating sam prompts in 3d for zero-shot instance segmentation. In *2025 International Conference on 3D Vision (3DV)*, pages 1222–1232. IEEE, 2025. 2

[53] Jiahui Yang, Jason Jingzhou Liu, Yulong Li, Youssef Khaky, Kenneth Shaw, and Deepak Pathak. Deep reactive policy: Learning reactive manipulator motion planning for dynamic environments. *arXiv preprint arXiv:2509.06953*, 2025. 2, 3

[54] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 2

[55] Haoqi Yuan, Ziye Huang, Ye Wang, Chuan Mao, Chaoyi Xu, and Zongqing Lu. Demograsp: Universal dexterous grasping from a single demonstration. *arXiv preprint arXiv:2509.22149*, 2025. 2

[56] Dingyuan Zhang, Dingkang Liang, Hongcheng Yang, Zhikang Zou, Xiaoqing Ye, Zhe Liu, and Xiang Bai. Sam3d: Zero-shot 3d object detection via segment anything model. *arXiv preprint arXiv:2306.02245*, 2023. 2

[57] Yuchen Zhou, Jiayuan Gu, Tung Yen Chiang, Fanbo Xiang, and Hao Su. Point-sam: Promptable 3d segmentation model for point clouds. *arXiv preprint arXiv:2406.17741*, 2024. 2

[58] Zhengxue Zhou, Xingyu Yang, Hao Wang, and Xuping Zhang. Digital twin with integrated robot-human/environment interaction dynamics for an industrial mobile manipulator. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5041–5047. IEEE, 2022. 2